Available online at www.sciencedirect.com

ScienceDirect

Procedia Technology 19 (2015) 584 – 590

Procedia Technology

8th International ConferenceInterdisciplinarity in Engineering, INTER-ENG 2014, 9-10 October 2014, Tirgu-Mures, Romania

Using WEB Services in SCADA Applications

Traian Turc*

* *Petru Maior" University of. Tg. Mures, 1 N. Iorga st., Tg. Mures, 540088, Romania*

Abstract

In order to access databases without have direct access, WEB Services can offer an easy way to benefits on data from databases. SCADA systems are built often around of databases. In such of case using WEB services is better solutions in order to avoid sophisticated way to access directly databases.

Peer-review under responsibility of "Petru Maior" University of Tirgu Mures, Faculty of Engineering

Keywords: SCADA, Wwb services, databases access, protocols

1. Introduction

1.1. SCADA Systems

SCADA systems usually rely on databases updated in real-time. The SCADA server, which is the main component, constitutes the interface between the monitored technological system and the SCADA applications. In other words, the SCADA server is responsible for facilitating the data flow between the technological process and the database. Similarly, the SCADA clients enable the human operators to read from or write to the database. Thus the main purpose of SCADA applications is to support the access to the database

While the operations on the database are expressed in the SQL language, the SCADA applications themselves are implemented as Web services. This implementation choice introduces an intermediary layer between the client and the server: the former does no longer need to be aware of the database structure, but rather expresses its information

* Corresponding author. Tel.: +40-265-233112.
E-mail address: traian.turc@ing.upm.ro

Peer-review under responsibility of "Petru Maior" University of Tirgu Mures, Faculty of Engineering
doi:10.1016/j.protcy.2015.02.083

needs as requests. The downside of this solution is the necessity of Web servers in addition to the standard SCADA servers. However, this overhead is compensated by the uniformity and ease of access to the information stored on the SCADA servers. This paper describes our solution in more detail.

1.2. Service-Oriented SCADA Software Architecture

Our software architecture incorporates a set of Web services that serve various components of the SCADA system. In other words, these components communicate over the Internet and make use of one or multiple Web services.

A service is a well-defined function, and is independent of its context or other services. Thus, Web services support easy and uniform, yet flexible access to the resources of a SCADA system.

On top of Web services, another paradigm that ensures uniformity is object-orientation

1.3. Web communications

Web communications follow a standard two-stage protocol: the client sends a request expressing an information need, and the server responds with the requested data. The two messages are expressed in a pre-defined format, known by both endpoints. For instance, SOAP (Simple Object Access Protocol) provides a useful specification for structured information transferred over networks. SOAP is one of the W3C (World Wide Web Consortium) accepted specifications

1.4. SOAP

SOAP specifies a two-part message format. The first part is an optional header containing metadata (e.g. for identification and encoding), while the second part contains the actual requested data described in XML (EXtensible Markup Language). SOAP messages are usually supported by the application layer of the networking stack, by protocols such as HTTP (Hypertext Transfer Protocol).

2. Web Services within SCADA Applications

2.1. Structure

The SCADA Web Services rely on SCADA servers, which are connected to the field devices through RTUs (Remote Terminal Units) and SCADA clients. The latter are situated either in the proximity of the SCADA servers, or further away - in which case connectivity is facilitated by the Internet. Figure 1 illustrates these ideas

The SCADA server communicates with the monitored technological system through RTUs. The collected data are stored in a database, which is updated in real-time, and serves as a central repository for the information needs of the SCADA clients.

In order to make sure that the clients have an updated view over the data at every instant in time, data must be repeatedly sent. A naïve solution is for the client to make very frequent HTTP requests, so that a pair of HTTP messages is sent for every update. This is wasteful in bandwidth and puts a significant burden on the HTTP server. A better alternative is to use an additional services server, which accepts service calls from clients. Then, the communication mechanism works as follows. First, the client sends an HTTP request. The HTTP server responds with a description of the Web and a set of service calls that the client will repeatedly direct to the services server. In this way, the final Web pages displayed by the clients' browsers are dynamic and show the data trends in real time.

Let us consider, for instance, a SCADA system for monitoring a gas plant, in which the SCADA server acquires data from various components through RTUs. Among these components, the most significant are the test points and the actuators. The data are stored in a database in real time. The SCADA clients can send requests to the Web servers to obtain HMI (Human-Machine Interaction) pages. As stated previously, once the client receives the page that also contains service calls, data will flow continuously from the services server towards the client.

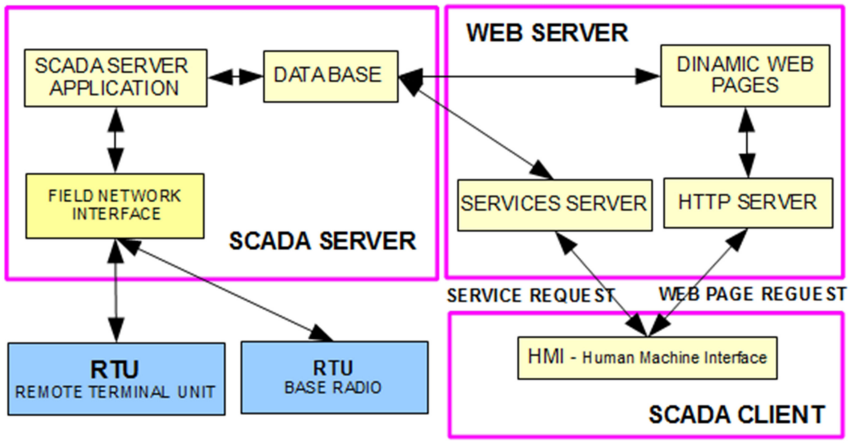

Fig. 1. SCADA architecture based on web services

The result is dynamic pages representing the HMIs of various installations within the monitored system. Figure 2 illustrates the HMI (Human-Machine Interaction) and shows some test points and actuators in a gas plant

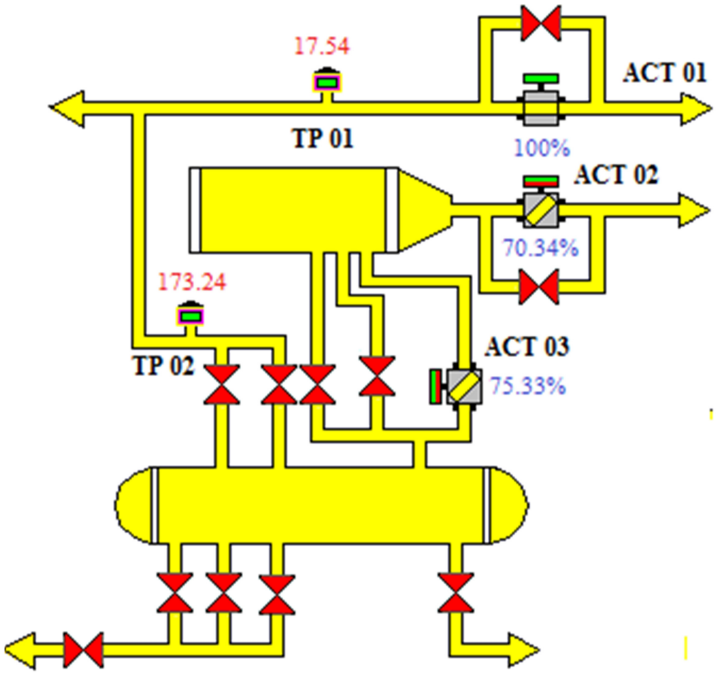

Fig. 2. gas plant HMI

Data regarding measurement points are stored in the "t_points" table, which is shown in Figure 3. The numerical figures refer to aspects such as the current, minimum or maximum value, etc.

tp_id	val	v_min	v_max	tp_cod
1	17.54	10.00	100.00	TP 01
2	173.24	10.00	200.00	TP 02
3	287.23	100.00	500.00	TP 03
4	45.22	0.00	100.00	TP 04
5	30.33	0.00	25.00	TP 05
6	334.89	0.00	250.00	TP 06
7	8.00	0.00	10.00	TP 07

Fig. 3. test points table

Data regarding actuators are stored in the "actuators" table, shown in Figure 4. The table contains numerical values for position, torque, etc.

act_id	act_cod	position	torque
1	ACT 01	100	115.33
2	ACT 02	70.37	104.25
3	ACT 03	75.33	140.75
4	ACT 04	67.80	10.55
5	ACT 05	60.00	138.00
6	ACT 06	99.99	0.00
7	ACT 07	76.00	45.00
8	ACT 08	74.54	137.69
9	ACT 09	100.00	0.00
10	ACT 10	0.00	0.00
11	ACT 11	67.33	85.88
12	ACT 12	58.54	39.77
13	ACT 13	0.00	0.00
14	ACT 14	0.00	0.00

Fig. 4. actuators table

2.2. Web Services

SCADA applications require access to data stored in SCADA servers. The access to the database is done either through classical queries, or through Web services. The latter provide an intermediate layer between the client and the server, ensuring uniform and easy access to data. For instance, Figure 5 shows a Web service that allows clients to read the values of the test points, or the actuator parameters.

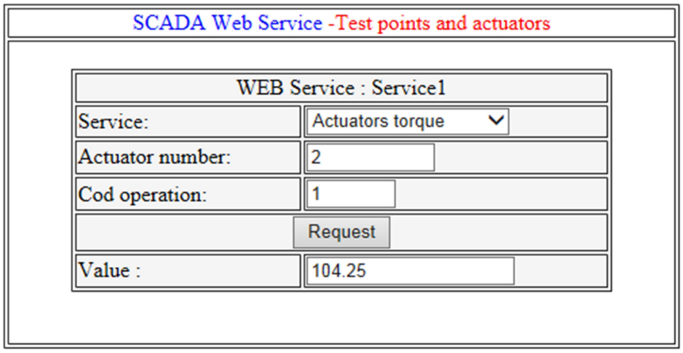

Fig. 5. Web service

3. Message Format

Both the requests and responses that flow between the server and its clients follow one of the W3C standards, namely SOAP

3.1. Requests

Requests contain the name of the service to be called and the necessary parameters. This information is structured in XML, according to SOAP. Figure 6 captures a client request to read data using Web Service 1

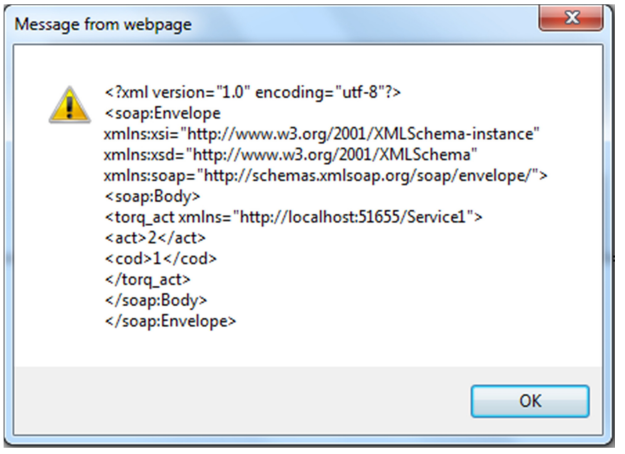

Fig. 6. Web service request

3.2. Responses

Responses contain the requested data in XML format, according to SOAP. Figure 7 shows the message returned by the request in Figure 6, which contains the result obtained by running Web Service 1

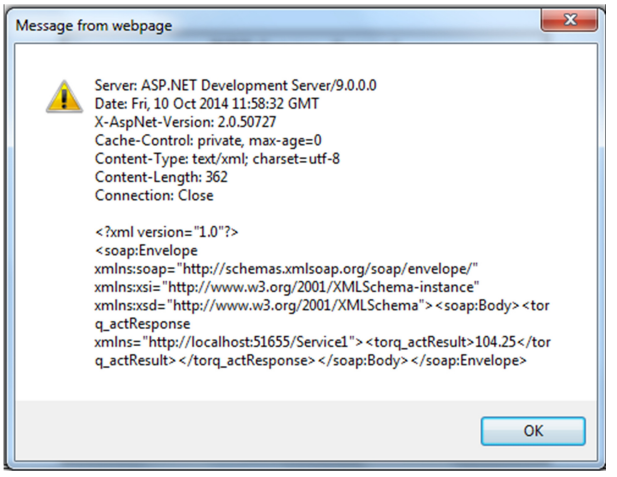

Fig. 7. Web Service response

4. Using Web Services to Directly Access Data Stored by SCADA Servers

Most SCADA architectures rely on a database. The purpose of this database is two-fold: first, it serves as a storage mechanism; second, it acts as a central repository that can be interrogated by clients to satisfy their information needs. Note that concurrent access issues are automatically solved by the database management system. Despite such convenient aspects of a database, it has a major downside: queries are slow and cannot serve well speed-critical applications. Therefore, one alternative is to eliminate databases completely and not to store any data on the SCADA server. Rather, this component becomes merely a forwarding entity

4.1. The Architecture of Web-based SCADA Services

Direct Web services are particularly useful when access speed is of uttermost importance. The high access speed of our system is due the direct communication between the SCADA clients and the server. In other words, the intermediary layer of the database is eliminated. Figure 8 shows our architecture

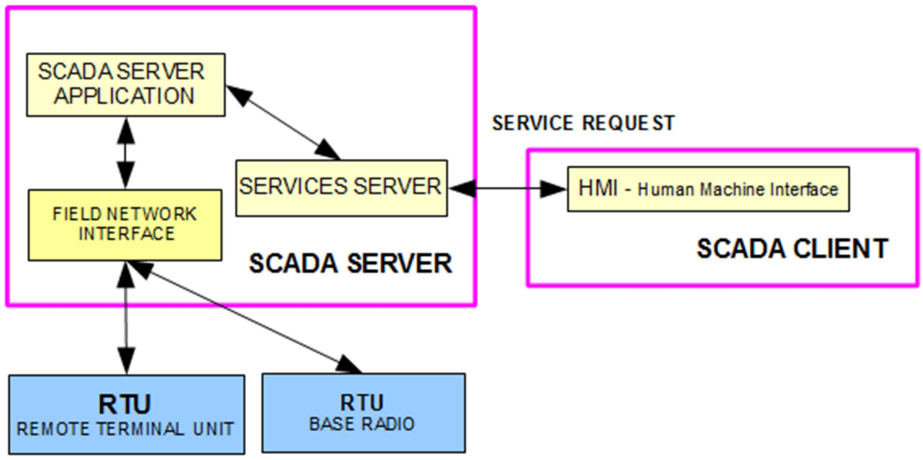

Fig. 8. direct access using Web services

4.2. Client Service Calls

In this architecture, the SCADA clients make direct requests to the SCADA server. As mentioned in section 2.1, the SCADA server must incorporate a services server. The acquired data are not stored in a centralized database anymore; this functionality is shifted to the clients. The server acquires data from RTUs with a certain frequency and sends it over to the clients that request it. In the lack of a centralized database, the access speed increases significantly

Conclusions

Using Web services within SCADA applications enables uniform access to the resources stored by the SCADA servers. Clients do not have to query databases, but can call Web services instead. This provides stronger data security, as the clients no longer possess the passwords to the database

When the speed of the data transfer is critical, it is better to deploy an architecture in which the clients are directly connected to the SCADA server. The downside of this approach is that there is no central repository, and each client must store its own data. Furthermore, any connection interruptions can cause significant data losses

References

[1] Damian DG Gessler, Gary S Schiltz, SSWAP: A Simple Semantic Web Architecture and Protocol for semantic web services „BMBioinformatics" 2009.
[2] Gligor, A., Turc, T., 2012. Development of a service oriented SCADA system, in "Emerging Markets Queries in Finance and Business", vol. 3, pp.256-261.
[3] Robles, R. J., Choi, M.-k., 2009. Assessment of the Vulnerabilities of SCADA, Control Systems and Critical Infrastructure Systems, in Int. Journal of "Grid and Distributed Computing", vol. 2, No. 2, Osvaldo Gervasi, Editor, SERSC, Republic of Korea, pp.27-34.
[4] Turc, T., Gligor, A., Dumitru, C., Morar, A., 2010. Development of extensible virtual instruments for SCADA applications, in Proc. of "2010 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)", vol. 3, pp.1-5.
[5] Turc, T., Gligor, A., 2011. Development Of Service Oriented Web-Based Scada Application, in "Scientific Bulletin of Petru Maior University of Tirgu Mures", vol. 8, nr. 2, Petru Maior University Press, pp.119-122.